Thu Dau Mot University Journal of Science
ISSN 2615 - 9635
journal homepage: ejs.tdmu.edu.vn

Determine the parameters of PID algorithm in liquid level control controlled by PLC s7 1500 using 3d virtual reality model

*by **Nguyễn Thành Đoàn** and **Phạm Hồng Thanh** (Thu Dau Mot University)*

***Article Info**: Received May 18th,2021, Accepted Nov. 25th,2021, Available online Dec. 15th,2021*
Corresponding author: thanhph@tdmu.edu.vn
https://doi.org/10.37550/tdmu.EJS/2021.04.260

ABSTRACT

It is very common to stabilize the preset value (Wanted value) of analog signals such as temperature, pressure, weight, flow, speed in automatic control. However, these control objects often have some problems such as overshooting, taking a long time to bring the system to a steady value, and large errors. One of the most used systems to overcome these problems is the PID, which is a preset stabilizing system with a quick function that returns the system to the set value in a short time without overshooting. error is close to zero. However, determining the scale parameters Ki, integral Kp, and differential Kd for the system to work optimally is a problem that needs to be studied. This paper presents how to accurately determine differential, integral, and scale coefficients according to 3D virtual reality model. Used a lot in simulation modeling for training and practical applications.

***Keywords**: PID, PI, PD, system stability, PLC S7-1200, virtual reality PID, method of determining PID parameters*

1. Introduction

In industrial automation control system, PID algorithm is widely used to stabilize physical quantities. However, each different system has different responses, so no PID model is used for all, so for each system we have to find out the proportional, integral and micro-control parameters. classification… most suitable for each system. When the system changes hardware, the PID control parameters must also change, the new control responds well with an error close to zero (Ang et al., 2005).

In order to build a PID control system that meets the constraints of error, overshoot and

setup time, a lot of algorithm tools and software support such as automatic detection of PID parameters according to algorithms are required. intelligent (Ang et al., 2005). The authors of this paper also only followed a specific physical model, still having large errors and overshoots as well as a longer establishment time. To overcome noise in the PID system, the authors (Han, 2009) used an active anti-noise algorithm in ADRC automatic control (Soe & San, 2019).

The article that finds out the optimal parameters of a PID system of a virtual model is done according to the following steps:

– Programmable PID PLC S7 1500 stably control water level with linear valve

– Build a 3D virtual model including the tank containing the supply and discharge linear control valves

– Coordinate the S7 1500 PLC program in Tia Portal software and 3D model in Factory IO to detect the parameters in the PID algorithm to control the stable system, fully meeting the requirements of an optimal control system (Ang et al., 2005; Linkens & Abbod, 1992; Sartika et al., 2019; Soe & San, 2019).

2. Theoretical Basis

2.1. Principle Control

Linear valve opening angle control by means of Level control system is a simple example of controlled system. A sensor measures the water level height and transfers the value to a controller. The controller compares the current water level with a setpoint and calculates an output value (manipulated variable) for level control (Ang et al., 2005).

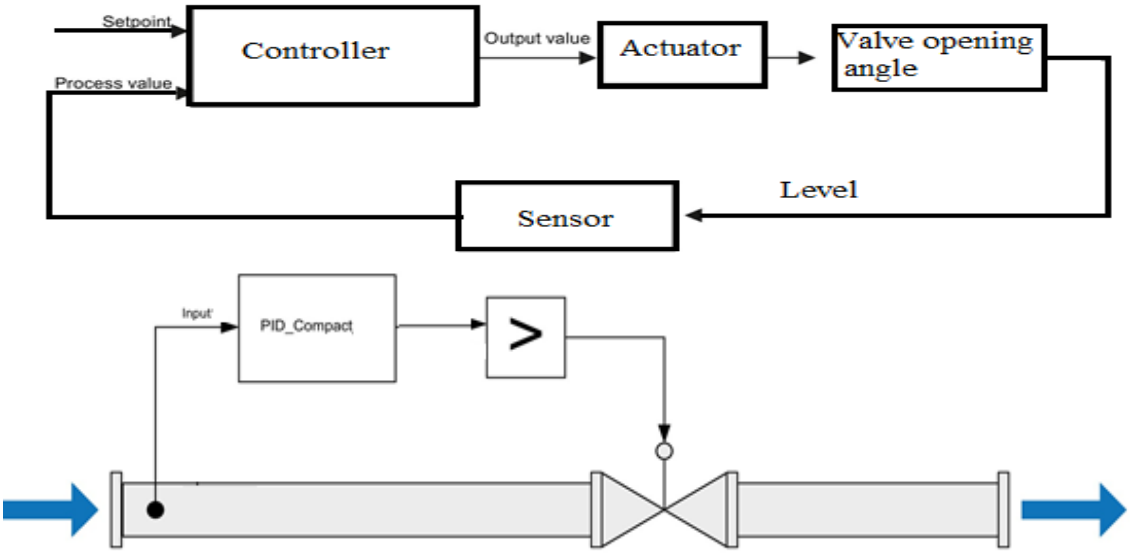

Figure1. *The valve is controlled with the output value of PID_Compact in I/O format (parameter Output_PER) by writing the program tag ActuatorInput*

2.2. PID Algorithm

PID controller with anti-windup and weighting of the proportional and derivative actions. The PID algorithm operates according to the following equation (Sartika et al., 2019):

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

Symbol	Description
y	Output value of the PID algorithm
K_p	Proportional gain
s	Laplace operator
b	Proportional action weighting
w	Setpoint
x	Process value
T_I	Integral action time
T_D	Derivative action time
a	Derivative delay coefficient (derivative delay T1 = a × T_D)
c	Derivative action weighting

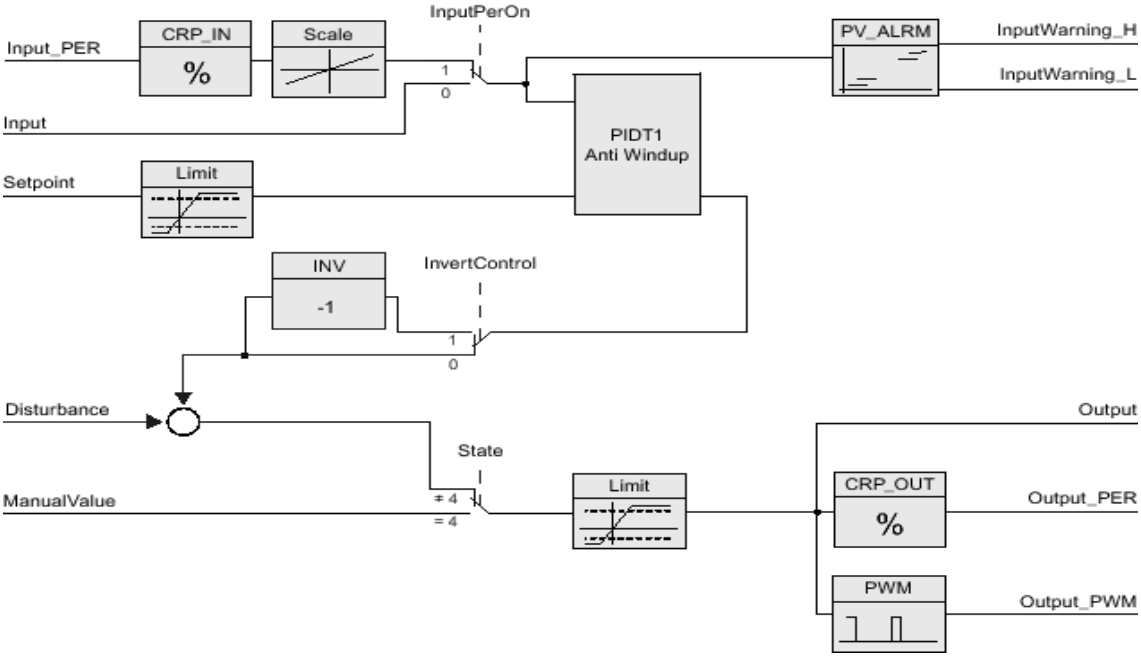

***Figure 2**. PID Algorithm logic programed in PLC S7 1500* (Soe & San, 2019)

2.3. PID Turning

Pretuning

The pretuning determines the process response to a jump change of the output value and searches for the point of inflection. The PID parameters are calculated from the maximum rate of rise and dead time of the controlled system. You obtain the best PID parameters when you perform pretuning and fine tuning (Al Andzar & Puriyanto, 2019; Sartika et al., 2019).

Pretuning requirements:

• Inactive (State = 0), manual mode (State = 4), or automatic mode (State = 3)

ManualEnable = FALSE

• Reset = FALSE

• The process value must not be too close to the setpoint.

|Setpoint - Input| > 0.3 * | Config.InputUpperLimit - Config.InputLowerLimit| and

|Setpoint - Input| > 0.5 * |Setpoint|

• The setpoint and the process value lie within the configured limits.

The more stable the process value is, the easier it is to calculate the PID parameters and the more precise the result will be. Noise on the process value can be tolerated as long as the rate of rise of the process value is significantly higher compared to the noise.

The setpoint is frozen in the CurrentSetpoint tag. Tuning is canceled when:

• Setpoint > CurrentSetpoint + CancelTuningLevel orSetpoint < CurrentSetpoint - CancelTuningLevel

Before the PID parameters are recalculated, they are backed up and can be reactivated with LoadBackUp.

The controller switches to automatic mode following successful pretuning. If pretuning is unsuccessful, the switchover of the operating mode is dependent on ActivateRecoverMode.

Fine tuning

Fine tuning generates a constant, limited oscillation of the process value. The PID parameters are recalculated based on the amplitude and frequency of this oscillation. PID parameters from fine tuning usually have better master control and disturbance characteristics than PID parameters from pretuning. You obtain the best PID parameters when you perform pretuning and fine tuning.

PID_Compact automatically attempts to generate an oscillation greater than the noise of the process value. Fine tuning is only minimally influenced by the stability of the process value.

The setpoint is frozen in the CurrentSetpoint tag. Tuning is canceled when:

• Setpoint > CurrentSetpoint + CancelTuningLevel or Setpoint < CurrentSetpoint - CancelTuningLevel

Before the PID parameters are recalculated, they are backed up and can be reactivated with LoadBackUp.

Requirements for fine tuning:

• No disturbances are expected.

• The setpoint and the process value lie within the configured limits.

• ManualEnable = FALSE

• Reset = FALSE

• Automatic (State = 3), inactive (State = 0) or manual (State = 4) mode

Fine tuning proceeds as follows when started from:

• Automatic mode (State = 3)

Start fine tuning from automatic mode if you wish to improve the existing PID parameters through tuning.

PID_Compact controls the system using the existing PID parameters until the control loop has stabilized and the requirements for fine tuning have been met. Only then will fine tuning start.

• Inactive (State = 0) or manual mode (State = 4)

If the requirements for pretuning are met, pretuning is started. The determined PID parameters will be used for control until the control loop has stabilized and the requirements for fine tuning have been met.

If the process value for pretuning is already too near the setpoint or PIDSelfTune.TIR.RunIn = TRUE, an attempt is made to reach the setpoint with the minimum or maximum output value. This can produce increased overshoot.

Only then will fine tuning start.

The controller switches to automatic mode following successful fine tuning. If fine tuning is unsuccessful, the switchover of the operating mode is dependent on ActivateRecoverMode (Burinskiene et al., 2018; Kosfeld, 1998; Koshal et al., 2019; Soe & San, 2019).

The "Fine tuning" phase is indicated with PIDSelfTune.TIR.State

3. Hardware And Software Components

This paper used Tia Portal V16 software programmed for PLC S7 1500 (CPU 1511-1PN), connected to Factory IO water level stabilization model by PID algorithm. Block Diagram to connected between components was shown as Figure 3.

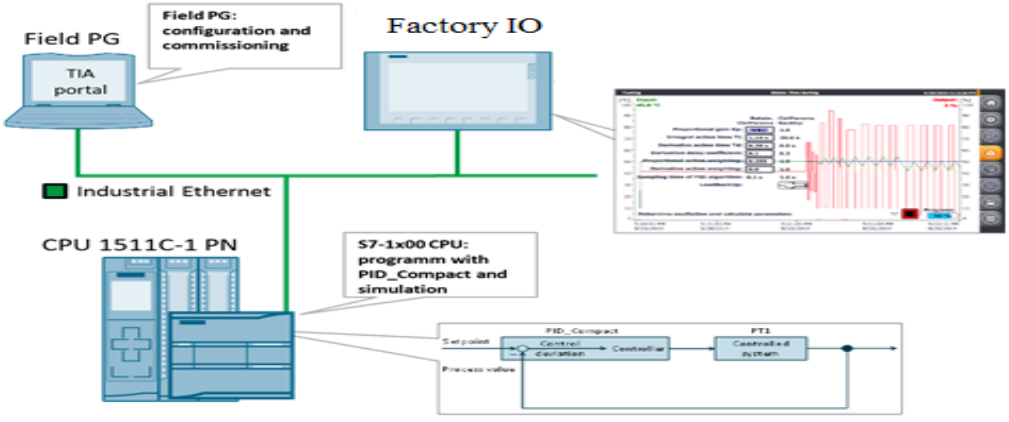

Figure 3*. Block Diagram connected between components*

4. Technical Results and Discussion

PID algorithm default parameters could be set up and programed in TIA Portal software as Figure 4.

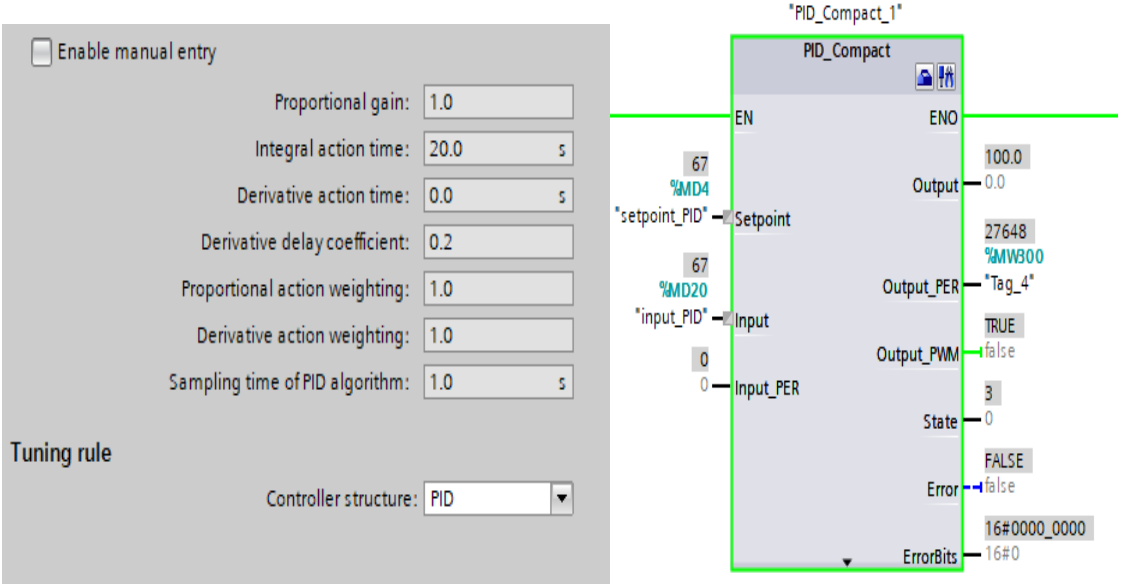

Figure 4

After time for the system run and found the best parameters it will propose these value of Parameters after detecting preturning and fine turning as we presented in Figure 5 below

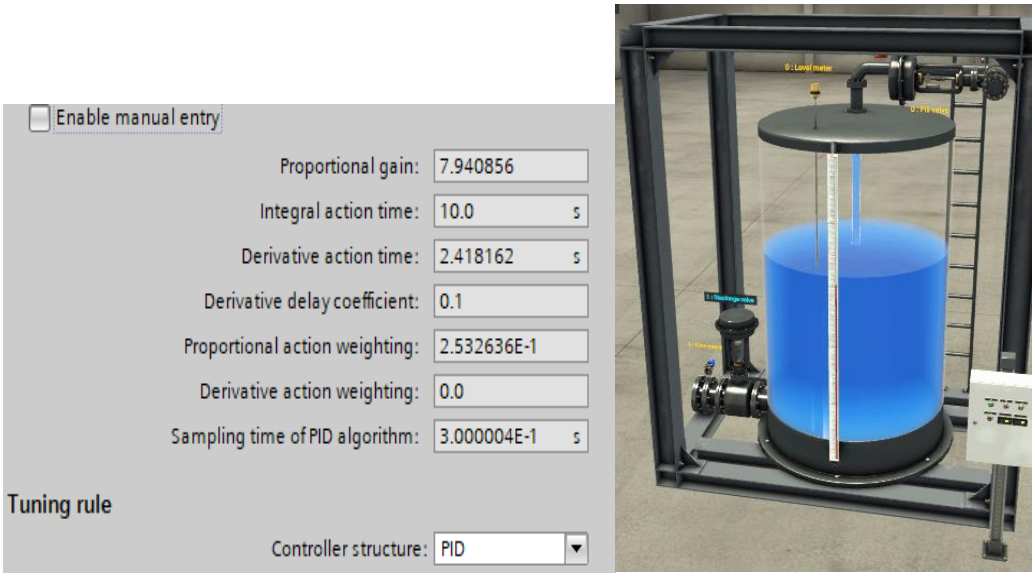

***Figure 5**. PID parameters after turning and the Virtual 3D system by Factory IO*

So, after programming the PLC S7 1500 in Tia portal V16 software connecting the 3D Factory IO virtual model, we can find the optimal parameters of the PID algorithm. This result is used in training students in automation and can be simulated to get PID parameters for the actual system after we build a virtual model with supporting software.

References

Al Andzar, M. F., & Puriyanto, R. D. (2019). PID Control for Temperature and Motor Speed Based on PLC. *Signal and Image Processing Letters*, *1*(1), 7-13. https://doi.org/10.31763/simple.v1i1.150

Ang, K. H., Chong, G., & Li, Y. (2005). PID control system analysis, design, and technology. *IEEE Transactions on Control Systems Technology*, *13*(4), 559-576. https://doi.org/10.1109/TCST.2005.847331

Burinskiene, A., Lorenc, A., & Lerher, T. (2018). A Simulation Study for the Sustainability and. *International Journal of Simulation Modelling*, 17, 485-497.

Kosfeld, M. (1998). Warehouse design through dynamic simulation. *Winter Simulation Conference Proceedings*, 2, 1049-1053. https://doi.org/10.1109/wsc.1998.745852

Koshal, A., Natarajarathinam, M., & Johnson, M. D. (2019). *Workforce training and industry 4.0 adoption in warehouses at SMEs. ASEE Annual Conference and Exposition, Conference Proceedings*. https://doi.org/10.18260/1-2--33669

Linkens, D. A., & Abbod, M. F. (1992). Self-organising fuzzy logic control and the selection of its scaling factors. *Transactions of the Institute of Measurement & Control*, *14*(3), 114-125. https://doi.org/10.1177/014233129201400301

Sartika, E. M., Sarjono, T. R., & Saputra, D. D. (2019). Prediction of PID control model on PLC. *Telkomnika (Telecommunication Computing Electronics and Control)*, *17*(1), 529-536. https://doi.org/10.12928/TELKOMNIKA.v17i1.11589

Soe, Y. Y., & San, P. E. (2019). Pid closed-loop control analysis for automation with siemens plc using tia *13*(2), 200-208.